REMARKS

Claims 1-27 are pending in the present application. Claims 1, 14, 16 and 22 are the independent claims. Claims 14, 16 and 22 were amended to more clearly recite the invention. No new matter has been added.

In the Official Action, dated August 4, 2003, the Title of the invention was objected to for allegedly being non-descriptive. The Abstract was objected to as to matters of form. The drawings are objected to for mislabeling Figures alleged to be admitted prior art. Claims 14-21 were rejected under 35 U.S.C. § 101, for allegedly being drawn to non-statutory subject matter. Claims 1-10, 12-14 and 16-27 were rejected under 35 U.S.C. § 102(b) as allegedly anticipated by U.S. Patent No. 5,870,763 (Lomet). Claims 11 and 15 were rejected under 35 U.S.C. § 103(a) as allegedly obvious over Lomet in view of U.S. Patent No. 6,513,019 (Lewis).

Objections to the Specification

The Title was objected for being non-descriptive. The Title of the invention has been amended herein to be more descriptive. In this regard, the invention provides a framework for applications to register their dependencies to other applications and objects, whereby the dependencies can be communicated to backup or recovery services for protecting against system failures and crashes. The Title has thus been amended to be in conformance with this general description of the invention. Withdrawal of the objection to the Title is respectfully requested.

The Abstract was objected to on grounds of lack of brevity. The Abstract has been amended to include 150 words or less, and to include only one paragraph. Accordingly, withdrawal of the objection to the Abstract is respectfully requested.

Objection to the Drawings

Figs. 1, 2, 3A and 3B were objected to for lacking the label "Prior Art" since the Office Action maintains "only that which is old is illustrated." Applicants respectfully disagree with this characterization, i.e., Applicants respectfully disagree that Figs. 1, 2, 3A and 3B were included in the present application to describe only that which is old. To the contrary, Applicants included Figs. 1, 2, 3A and 3B as illustrative of the types of computing environments, computing systems, and computing objects to which the invention directly applies.

For instance, computer 20 and network 14 of Figs. 1 and 2 illustrate exemplary, non-limiting computing environments in which the invention may be deployed. Similarly, Figs. 3A and 3B illustrate exemplary computing scenarios to which the invention applies, i.e., Figs. 3A and 3B are exemplary hierarchies of state dependencies, reflecting vertical and horizontal aspects of application dependency information, to be captured in storage in accordance with the invention for use in connection with a service. Since Figs. 1-3B serve to illustrate the invention's application and deployment environment, Applicants respectfully submit that Figs. 1-3B illustrate subject matter related to the present invention, not subject matter which is old prior art. Withdrawal of the objection to the drawings is respectfully requested.

Rejections under 35 U.S.C. § 101

Claims 14-21 were rejected under 35 U.S.C. 101 because the claimed invention is purportedly directed to non-statutory subject matter.

In view of the amendments herein reciting that the data structure and application programming interface of claims 14 and 16, respectively, are embodied in computer readable media and in view of the additional functional recitation describing the useful, concrete and tangible result" of the data structure of claim 14, Applicants respectfully submit that the outstanding rejection under 35 U.S.C. 101 has been rendered moot. See, e.g., *State St. Bank & Trust Co. v. Signature Fin. Group*, 149 F.3d 1368 (Fed. Cir. 1998) ("the mere fact that a claimed invention involves inputting numbers, calculating numbers, outputting numbers, and storing numbers, in and of itself, would not render it nonstatutory subject matter, unless, of course, its operation does not produce a 'useful, concrete and tangible result.'"). See also *In re Lowry*, 32 F.3d 1579 (Fed. Cir. 1994) (The Board found that the claims directed to a memory containing stored information, as a whole, recited an article of manufacture and thus were statutory subject matter.).

For the reasons set forth above, Applicants submit that claims 14-21, as amended, recite patentable subject matter and that claims 14-21 are in conformance with the requirements set forth in MPEP § 2106. Applicants therefore respectfully request reconsideration of the rejection of claims 14-21 on the basis of 35 USC § 101.

Summary of the Invention

The present invention relates to the backup or restoration of aspects of a computer system. More particularly, the present invention relates to a method and system for storing information about dependencies among applications and objects, so that a computer system may efficiently return to a pre-crash state in connection with the dependency information.

When an application is no longer responding, it is usually desirable from a system standpoint to terminate only the application that is not responding, or to find an application dependency that may be causing the program to freeze its response. Indeed, it is often the case that a programmer of **Application A can not anticipate all of the combinations and permutations of other applications B, C, D, etc. that will be installed on a given computer system for running application A**. **A state of operation of an application A may depend upon certain other applications, storage components, processes and files that are in application A's system space for proper execution, and these dependencies may be changing dynamically**. There may be embedded dependencies that must be unraveled in an order that is determined by those dependencies in order to perform a backup operation. In consideration of those dependencies, the invention provides a way of handling backup operations **taking into account the state of an application A and its vertical or horizontal dependencies (i.e., hierarchically organized dependencies) on other applications or objects**. See, e.g., page 4, lines 3-18, and Figs. 3A-3B for exemplary application dependencies.

In various embodiments, the invention provides a way for an application to register its dependencies via a common software agent, which in turn can coordinate backup services based

upon application dependencies. As illustrated in Fig. 3B, an application 1 may have knowledge of its own dependencies, i.e., to database 400d and database 400c, but application 1 does not have knowledge of the dependencies of application 2. Accordingly, the common software agent 510 (described and illustrated in connection with Fig. 5) provides an intelligent view over all of the dependencies of the applications executing on the system.

Lomet

Lomet relates to a logging recovery technique that captures the state of an application with respect to logging operations. In consideration that the application states of executing applications of a system are generally not captured (See, e.g., Col. 3, line 58 to Col. 4, line 4), with Lomet, in the event of a system failure, the database computer system begins with the stable database state and replays the stable log to redo certain logged application operations. The database computer system redoes a logged application operation if its state ID is later in series than the state ID of the most recently flushed or already partially recovered application state. See, e.g., Col. 6, lines 23-29.

Lomet discloses that application operations are defined that produce transitions between application states. The **application state is treated as a single object** that can be atomically flushed to the stable database. See, e.g. Col. 6, lines 12-22. These operations are immediately entered into the volatile log, and subsequently posted to the stable log. See, e.g., Col. 6, lines 7-11. The state of an application can change as a result of application execution, interaction with the resource manager, interaction with each other, and interaction with the terminal.

Rejections under 35 U.S.C. §§ 102, 103

Claims 1-10, 12-14 and 16-27 were rejected under 35 U.S.C. § 102(b) as allegedly anticipated by Lomet. Claims 11 and 15 were rejected under 35 U.S.C. § 103(a) as allegedly obvious over Lomet in view of Lewis. The outstanding rejections to the claims under 35 U.S.C. §§ 102(b), 103(a) are respectfully traversed.

While Lomet discloses to capture the state of an application with respect to pre-defined logging operations (e.g., read, write operations), dependencies associated with the application state discussed in Lomet are fundamentally different than the dependency information among applications and objects captured by the present invention.

As described at Col. 13, lines 24-39, with the logging system of Lomet, with an execute operation of the application, for instance, **the application executes instructions internal to the application**, whose effects are hidden from the external world. **This execution transforms the application from a state A.sub.1 to a state A. sub.2.** Following this execution, the application calls to the resource manager. The resource manager logs the application Execute operation Ex(A. sub.1) denoting the transformation of application A from state A, to state A.sub.2 to the volatile log for subsequent posting by the log manager into the stable log. **As denoted in FIG. 6, the resource manager logs the application identifier A, its state ID 2, and the execute operation Ex that resulted in the application state A.sub.2.** Thus, it is clear in this example, that an operation purely internal to the application creates a change in state of the application.

With Lomet, the application state transformations between interactions with a resource manager are logged as operations in the volatile log 68. At recovery, these logged state transformation operations are replayed, with the effect being that the hidden internal changes leading to each logged state are repeated. See, e.g., Col. 12, lines 45-50.

Thus, execution of an application 60 is characterized as a series of loggable atomic operations whose replay can recover the application. To capture application execution as a series of loggable operations, the computer system 50 treats the code execution between calls in the application as the log operation. See, e.g., Col. 12, line 66 to Col. 13, line 4.

Thus, **Lomet maps the application execution into loggable operations such that the operations can be expressed entirely in terms of the state of the application state**. For the execute operation, for example, the application begins in one state and is transformed to another state by internal executions of the application. To the outside world, the execute operation can therefore be expressed as reading a first application state before the internal executions, and writing a transformed application state resulting from the internal executions. See, e.g., Col. 14, lines 36-43.

In contrast, with Applicants' invention, an application registers its **dependencies to other applications and objects** in the computing system with a common software agent, not its application state as it applies to the five logical operations of an application (execute, initiate, terminate, read and write). See, e.g., Col. 13, lines 13-14. With Lomet, only operations affecting the address space of an executing application are considered a change in state.

Accordingly, Lomet cannot be said to teach or suggest a method comprising "registering applications loaded in said computer system with an application dependency application programming interface (API) for communications of **application dependency information among applications**, a common software agent, a storage component utilized by said agent and a backup service" (claim 1), a data structure comprising "data representative of said at least one external dependency" (claim 14), an application programming interface (API) that "enables an agent to collect, store and package **information about dependencies among applications** in response to a request by a service, and thereafter delivers said application dependency information to said service for further processing by said service," (claim 16) or a computer system, comprising "an agent that functions according to communication protocols of an application programming interface (API) in said system for processing application dependency information**, wherein said application dependency information includes information about dependencies among applications executing on the system,** communicated to said API from said agent and for storing the application dependency information in said storage component" (claim 22).

Lewis was cited for its disclosure relating to extensible markup language (XML), but Lewis does not cure the above-identified deficiency of the root reference Lomet with respect to claims 1, 14, 16 and 22.

Claims 2-13, 15 and 21, 17-20 and 23-27 depend either directly or indirectly from claims 1, 14, 16 and 22, respectively, and are believed allowable for the same reasons. Accordingly, Applicants submit that claims 1-27 patentably define over Lomet and Lewis, taken alone or in

combination. Withdrawal of the rejection of claims 1-27 under 35 U.S.C. §§ 102(b), 103(a) is thus earnestly solicited.

CONCLUSION

Applicants believe that the present Amendment is responsive to each of the points raised by the Examiner in the Office Action, and submit that Claims 1-27 of the application are in condition for allowance. Favorable consideration and passage to issue of the application at the Examiner's earliest convenience is earnestly solicited.

Date: November 4, 2003

Thomas E. Watson
Registration No. 43,243

Woodcock Washburn LLP
One Liberty Place - 46th Floor
Philadelphia PA 19103
Telephone: (215) 568-3100
Facsimile: (215) 568-3439